Using Neural Networks for Sensor Validation

Duane L. Mattern and Link C. Jaw
Scientific Monitoring, Inc., Tempe Arizona

Ten-Huei Guo
Lewis Research Center, Cleveland, Ohio

Ronald Graham and William McCoy
Allison Engine Company, Indianapolis, Indiana

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the Lead Center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:
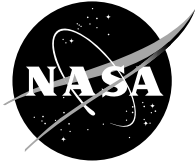
- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA's counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that complement the STI Program Office's diverse offerings include creating custom thesauri, building customized data bases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at *http://www.sti.nasa.gov*

- E-mail your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at (301) 621-0134

- Telephone the NASA Access Help Desk at (301) 621-0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076

Using Neural Networks for Sensor Validation

Duane L. Mattern and Link C. Jaw
Scientific Monitoring, Inc., Tempe Arizona

Ten-Huei Guo
Lewis Research Center, Cleveland, Ohio

Ronald Graham and William McCoy
Allison Engine Company, Indianapolis, Indiana

National Aeronautics and
Space Administration

Lewis Research Center

Available from

USING NEURAL NETWORKS FOR SENSOR VALIDATION

Duane L. Mattern and Link C. Jaw
Scientific Monitoring, Inc., Tempe, Arizona

Ten-Huei Guo
NASA Lewis Research Center, Cleveland, Ohio

Ronald Graham and William McCoy
Allison Engine Company, Indianapolis, Indiana

Abstract

This paper presents the results of applying two different types of neural networks in two different approaches to the sensor validation problem. The first approach uses a functional approximation neural network as part of a nonlinear observer in a model-based approach to analytical redundancy. The second approach uses an auto-associative neural network to perform nonlinear principal component analysis on a set of redundant sensors to provide an estimate for a single failed sensor. The approaches are demonstrated using a nonlinear simulation of a turbofan engine. The fault detection and sensor estimation results are presented and the training of the auto-associative neural network to provide sensor estimates is discussed.

Nomenclature

AANN	Auto-Associative Neural Network
ALT	Aircraft altitude, (feet)
CVGFB	Compressor Variable Geometry FeedBack
CVGMA	CVG MilliAmpere command signal
DPBLD	Delta Pressure between the compressor discharge and the bypass duct
DTAMB	Temperature variation from standard day temperature, (deg F)
FADEC	Full Authority Digital Engine Control
FDIA	Fault Detection, Isolation, & Accommodation
ITT	Interstage Turbine Temperature (deg F)
MMVFB	Main Metering Valve position FeedBack
P25	Low pressure compressor inlet pressure
P3	Compressor discharge pressure, burner inlet
PLA	Power Lever Angle (thrust demand)
T25	Compressor inlet temperature (deg F)
T3	Temperature at burner inlet, (deg R)
WF	Fuel flow (lbm/hour)
WFMA	Fuel Flow MilliAmpere command signal
XM	Aircraft Mach number
XNL	Fan rotor speed, (rpm)
XNH	Core rotor speed, (rpm)

Introduction

Safety and reliability are key design issues in turbine engines. Methods such as analytical redundancy for handling online faults can be used to increase an aircraft's reliability. Analytical redundancy has been demonstrated on a turbofan engine in reference [1]. This approach used an online nonlinear model of an engine to provide estimates for failed sensors. The model was tuned to closely match the steady state and dynamic response of the actual engine. This demonstration required a relatively high fidelity and highly tuned real-time engine model. In reference [2] a bank of Kalman filters was used to provide probabilistically weighted parameter estimates of measurements. This approach required a dither to disturb the system from a quiescent state in order to identify the system online. As an alternative, an auto-associative neural network was used for sensor validation of a rocket engine in reference [3]. This reference indicates that the neural network estimates of the sensor values could be used to replace failed sensor values in a feedback control system. The work presented here is a continuation of the work in reference [4] and is based on the work in [1,3,5].

In the following, the sensor validation problem is introduced and two approaches to the problem are presented: a model-based approach using a nonlinear observer, and an auto-associative neural network. The nonlinear observer uses neural networks to model the variation of the system with the operating point. The auto-associative neural network (AANN) approach is trained to be able to estimate a variable from a set of analytically redundant measurements when the sensor corresponding to that variable is faulty. The AANN provides a nonlinear principal component analysis and data dimensionality reduction via the funneling structure of the neural network. These two approaches are demonstrated on a model of a turbofan engine. Results are presented showing the overall system response during simulated sensor faults. The difficulty in training the AANN to provide estimates when provided erroneous information is also discussed.

The Sensor Validation Problem

While we will apply the approaches in this paper to nonlinear systems, we will use the following linear system to illustrate the concept of sensor validation:

$$\dot{x} = Ax + Bu \qquad (1)$$
$$y = Cx \qquad (2)$$

x is the system state vector, u is the system input vector, and y is the system output vector. A, B, and C are system matrices of appropriate dimension. The "m" outputs or measurements, y, are a linear function of the "n" state variables, x. Assuming that the A,B, and C matrices include models of the actuators and sensors, then y contains the sensed outputs used by the control system to generate the actuator commands, u. The sensor validation problem can be posed by the following three questions:

1) In a real physical system where hardware failures can and do occur, how does one *detect* when the information provided to the control system through the measurement vector, y, is incorrect?
2) Once it is known that something is wrong with the information presented in y, how does one *isolate* the source of the problem?
3) Once it is known where the problem is, (which sensor has failed), how does one *accommodate* the problem?

While faults occur in devices other than the sensors, (a component fault for example), the sensors must be validated first, prior to addressing any system components, since all information is obtained through the sensors. We are concerned only with sensor validation in this paper. The three steps to the sensor validation are Fault Detection, Isolation, and Accommodation (FDIA).
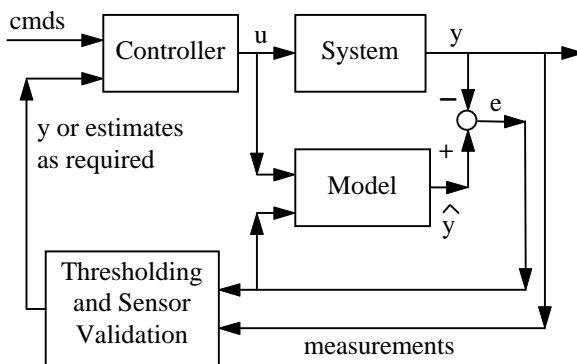

Figure 1   Block Diagram of Model-Based Approach to Sensor Validation

A Model-Based Approach to Sensor Validation

One approach to sensor validation is to include an onboard model of the system as part of a real-time diagnostics system. In this approach the output of the model is used to validate the sensed information. The absolute value of the difference between the model outputs, $\hat{y}$, and the real system measurements, y, is the error, e, which is fed back to the threshold scheme as shown in Figure 1. With a model-based method the violation of a single threshold isolates the fault to a single measurement. Then the estimated value can be used in place of the measured value. The onboard model is the heart of the system.

An onboard model was used in references [1,6,7]. In reference [6], a piecewise linear model of an engine is used in conjunction with a Kalman filter to provide an estimate of engine damage. Reference [7] used an online model as part of a control system. The main issue in reference [7] is the ability to provide estimates of unmeasurable variables, thrust and stall margin, for two new control modes. Reference [8] used functional approximation neural networks to schedule the control variables used in a full envelope control design for a simulation of the J-85 turbojet engine.

There are two main issues to be addressed when using an online model. The first is accuracy. When dealing with a real manufactured product, there are variations from product to product because of manufacturing tolerances. There are also changes associated with aging, and wear and tear in engines that are in service for extended periods. These engine-to-engine variations complicate the fault detection method using thresholds. Attempts to take these deviations into account by online identification and adaptation contribute to the second issue: computational overhead. As in any mass produced product, cost is a consideration and the extra computational overhead of an online model or model and adaptation scheme can easily double the computations required. This also greatly contributes to the complexity of the software checkout and validation task.

An AANN Approach to Sensor Validation

The second method, the Auto-Associative Neural Network (AANN) approach to sensor validation, can be used when the information in the measurements is analytically redundant in the sense that if one measurement is lost, it can be replaced with an estimate from the remaining valid sensors. For the system described by equations (1,2), consider the case with m sensors such that m is greater than the size of the state

vector, n (m>n). An estimate of the state vector can be made from a subset of the outputs, y_s, as long as the state vector is observable from the subset of outputs, as follows in equation (3):

$$\hat{x} = (C_s^T C_s)^{-1} C_s^T y_s \qquad (3)$$

C_s contains the rows of C corresponding to y_s. Note that in going from y_s to $\hat{x}$ there is a reduction in the dimension of the data. This state estimate can be used to estimate the full output, $\hat{y}$:

$$\hat{y} = C\hat{x} = [C(C_s^T C_s)^{-1} C_s^T] y_s \qquad (4)$$

(Note: Superscript T indicates a matrix transpose)

If a non-zero direct feed-through term, D, appears in equation (2), it would read: y=Cx+Du. In this case it is possible to create a fictitious measurement y* that can be constructed as y*=y-Du, so the above technique is still valid since u is known. The point to be made using equation (4) is that the subset of redundant sensor measurements can be used to estimate any missing measurements as long as the state is observable from the remaining subset of valid sensors.

However, in comparison, a Kalman filter provides better noise filtering and excellent dynamic estimates of the outputs of a linear system. For nonlinear systems, the extended Kalman filter can be used, but it imposes a much greater computational burden because of the complexities involved with representing a nonlinear system with a family of piecewise linear models. An auto-associative neural networks can be used to extend the functionality of equation (4) to nonlinear systems. The reduction of the data dimension going from y_s to $\hat{x}$ can be thought of as a principal component analysis. The structure of the auto-associative neural network used for the nonlinear system considered here essentially resolves the principal components which are nonlinear functions of the measurements. These principal components are then used to estimate all of the sensed variables.
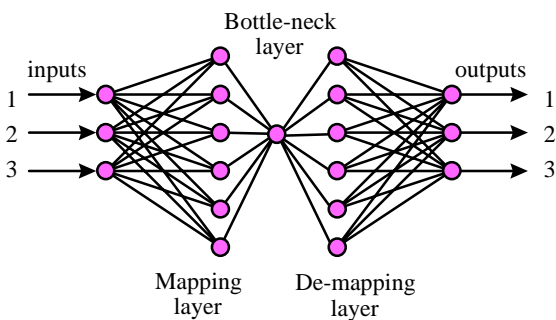

Figure 2 Structure of a Four Layer (Matlab notation)
3-6-1-6-3 Auto-Associative Neural Network

Auto-Associative Neural Networks (AANN)

An AANN is a feedforward network architecture with outputs which reproduce the network inputs. The architecture used here consists of two halves, the mapping layer (on the left in Figure 2) and the de-mapping layer. These halves are interconnected through the bottle-neck layer. The mapping layer compresses the data into a reduced order representation, eliminating redundancies and extracting the key features (principal components) in the data. The dimensionality reduction characteristics of this architecture are discussed in [5]. The de-mapping layer recovers the encoded information from the principal components.

The minimum number of nodes in the bottle-neck layer that will provide sufficient information for data recovery represents the degree of freedom of the data system. In the sense of a linear system, the bottle-neck layer must contain at least n nodes, were n is the size of a non-redundant state vector, (a minimal representation). The observability requirement still holds for nonlinear systems.

The AANN is trained in two steps. The first step trains the entire network using valid data. This requires the selection of the number of nodes in the various layers. The size of the bottle-neck layer is critical to obtain the desired effect of eliminating the redundancies in the measurements. The second step involves the modification of the training set to include false data in the sense of faulty measurements. The network is then retrained with a data set that includes erroneous information in order to learn how to filter the false information. There are various approaches to this second step, some of which include freezing specific weights within the AANN. We'll discuss two different approaches to this second training step next.

AANN Fault Training Approach 1   The first approach looks at the outputs of the bottle-neck layer as the weighted combination of the inputs. If one of the inputs is faulty, this fault will have a reduced effect on the bottle-neck layer because it is only one component of many data inputs. As a simple example, consider the case where this network structure is used for three measurements of the same temperature and the bottleneck layer is just one node. Then the mapping layer performs a weighted averaging of these temperature measurements. Faulty information in one sensor is then reduced to 33% of the original value because there are a total of three measurements used in the average calculation. Thus the bottleneck layer performs a weighted averaging.
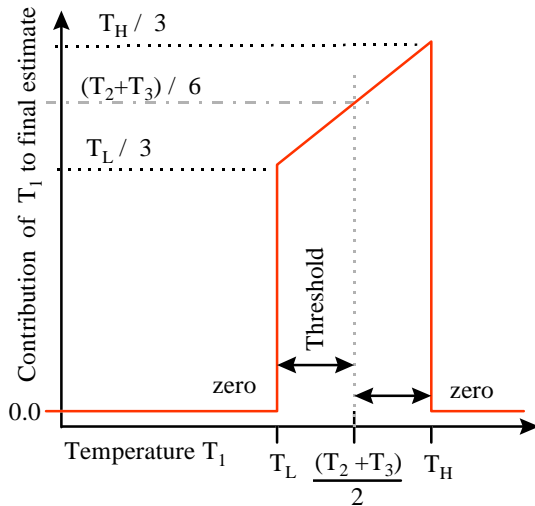
Figure 3 Functional Map of Logic
Training Approach 2

AANN Fault Training Approach 2   The second approach to retraining the mapping layer attempts to train the mapping layer to learn threshold logic [5]. Consider the previous example of three temperature measurements, T_1, T_2, T_3. If T_1 does not agree with the average of the other two temperatures, within some threshold (bounded by T_L and T_H in Figure 3), then the contribution of T_1 to the estimate of the temperature would be zero. We assume that T_2 and T_3 fall within some acceptable bounds for this example. Without T_1, the temperature estimate would be made from the average of T_2 and T_3, $(T_2+T_3)/2$. This voting logic is represented by a functional map as shown in Figure 3. The full functional map for this example would be four dimensional and has sharp contours associated with it. A neural network can be trained to approximate this functional map, but the map surface is complex and the training is tedious. Reference [5] discusses this method for training an AANN. The key advantage to this approach is that it allows the detection, isolation, and accommodation to be accomplished in one step. The key drawback of this approach is that the training is not trivial and a simple example as this sometimes is much easier to be coded in a standard software language.

In summary, the AANN approach to sensor validation trains a neural network to learn the relationships between a set of redundant sensors such that if one sensor is bad an estimate for that sensor can be obtained from the remaining valid sensors. Note that the neural network constitutes a model of a portion of the engine and as such it has the same accuracy, updating, and computational issues mentioned previously for more conventional models.

Example Using a Turbofan Engine Model

Examples of the two approaches to sensor validation are demonstrated using a nonlinear model of a turbofan engine consisting of the engine and control system (FADEC). The external variables are PLA, ALT, DTAMB, and XM. PLA is a pilot command input whereas ALT, DTAMB, and XM are considered external system inputs due to ambient flight conditions. The controller feedback variables are XNL, XNH, T25, ITT, MMVFB, and CVGFB. XNL, XNH, T25 and ITT are engine outputs whereas MMVFB and CVGFB are the actuator feedbacks of the fuel flow main metering valve and the compressor variable geometry position respectively. We also consider the additional measurements P25, P3, T3, WF, DPBLD which are used for the analytically redundant information needed in the AANN approach. While fuel flow, WF, is not normally directly measured, it can be obtained from another variable for this engine which will be shown. Figure 4 shows a block diagram of the closed-loop system. Note that MMVFB and CVGFB are actuator outputs, not engine outputs. These two variables are outputs of single input, single output servo loops. These servo loops are higher bandwidth inner-loops when compared to the engine outer loop variables. In this sense, they represent the system inputs, WFMA and CVGMA. In the following examples, we will first look at a classical model-based approach to analytical redundancy using a model-based, nonlinear observer to estimate the main metering valve feedback, MMVFB. Then we will use the AANN to detect faults in the two rotor speeds, and to provide estimates for the two rotor speeds, XNL and XNH.
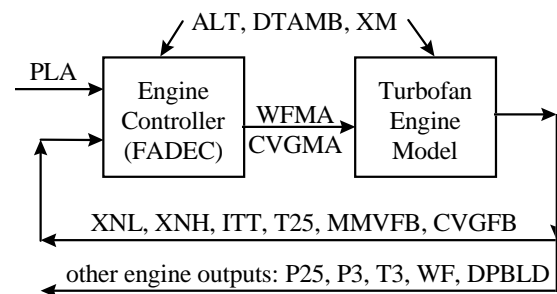

Figure 4 Schematic of Example Turbofan Model
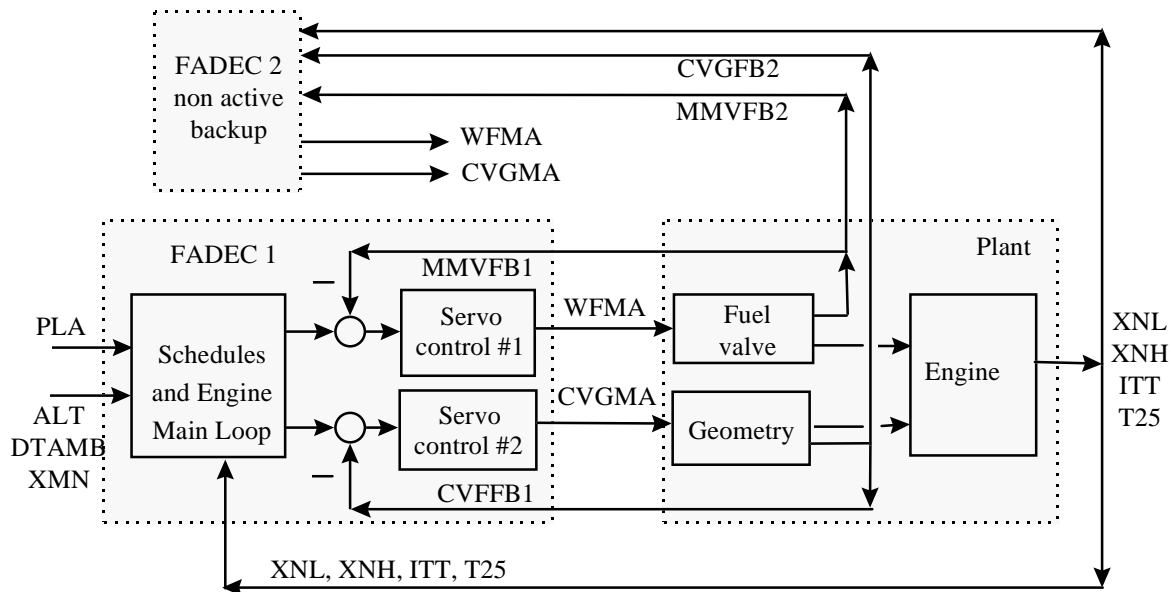Consisting of Separate Control and Engine Modules

Figure 5 Dual Redundant FADEC System Block Diagram

MMVFB Observer

MMVFB is the position feedback from the main metering fuel valve. Its purpose is to address the nonlinearities within the actuator to prevent "hunting" (oscillation) in the fuel control system. In a system with dual redundant FADEC's as shown in Figure 5, when the MMVFB sensor for FADEC #1 (MMVFB1) fails then FADEC #2 must take over the control of the engine assuming it still has a good MMVFB signal (MMVFB2). The MMVFB fault must be detected and the controller must be switched in one time sample in order to maintain the stability of the actuator loop because of the high bandwidth nature of this loop. Thus fast fault detection is critical to avoid a degraded mode.

One of the issues with a dual redundant FADEC system is that although differences between two redundant measurements can be detected, there is insufficient information to isolate a soft fault. A soft fault is a fault that does not cause a large, sudden change in the measurements that would be outside of the normal rate of change for that signal. A drifting measurement is an example of a soft fault. A tie breaking vote is required to isolate a soft fault and analytical redundancy can be used to provide a third

vote in a system with dual FADEC's. The advantage of having the third vote is that the sensor fault can be detected and isolated. This allows the valid sensor to be recognized and used, and permits the engine to operate normally without the need to switch to a degraded mode.

In this example a nonlinear observer is used to estimate the value of MMVFB. This particular observer uses two neural networks to estimate the core rotor speed, XNH, which is used to provide steady state correction to the MMVFB estimate as shown in Figure 6. This steady state correction prevents the accumulation of an error in MMVFB that could build up over time. Note that this correction term has an integral error term separate from the integral within the fuel valve dynamic model. A separate integral was used to avoid having to retune the deadband used within the fuel valve dynamic model (Figure 7). Not shown in Figure 6 is a range limit that was part of the integrator limit and windup protection logic that was included in this model. The dynamic behavior of MMVFB is estimated using a first order model that was identified from simulation data. This first order, nonlinear fuel valve dynamic model is shown in Figure 7 and contains a deadband nonlinearity.
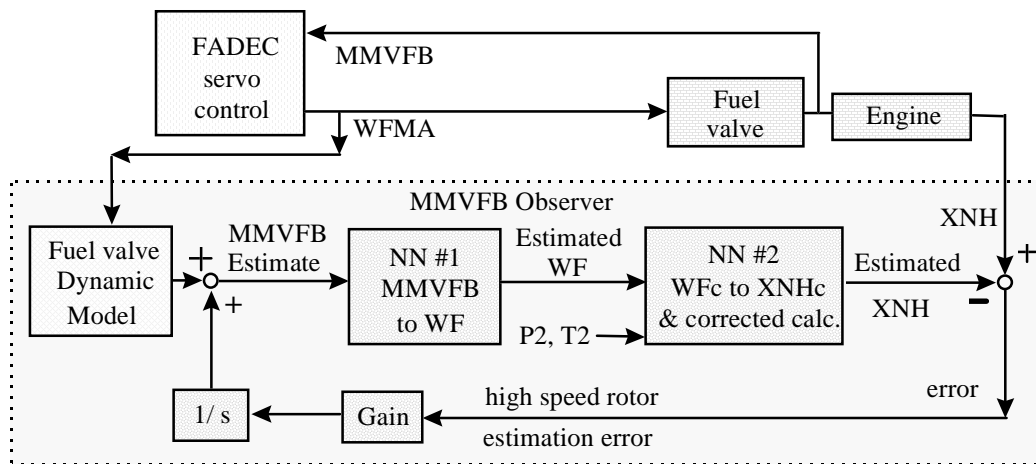
Figure 6 Block Diagram of MMVFB Nonlinear Observer with Neural Networks #1 and #2
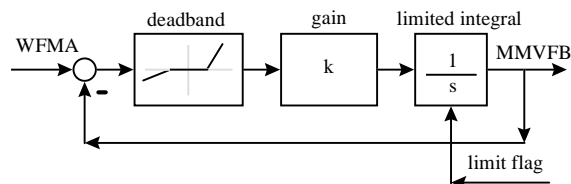

Figure 7 Fuel Valve Dynamic Model

Note that the deadband is not symmetric. It was tuned using transient responses from the closed loop engine simulation. The model of the engine was treated as a black box and the model source code was not available. This nonlinear, first order model comprised the dynamic response of the observer. The steady state correction path is based on the XNH estimate error. The XNH estimate is obtained through two neural networks which are described below.

Two single input, single output, neural networks are used in the MMVFB observer. Both networks are simple functional approximations that could have been accomplished with nonlinear curve fitting routines. The advantage of using neural networks for functional approximation is that the nonlinear functions used in the curve fitting do not have to be selected other than the nonlinearity used within the network node activation functions which is typically sigmoidal or some type of smooth "S" shaped function. The first network in Figure 6, (N.N. #1) generates a steady state estimate for fuel flow given MMVFB. The second network (N.N. #2) generates an estimate for corrected rotor speed (XNH_c) given corrected fuel flow, WF_c. Since no details were available regarding the control system schedules, the closed loop model was used to

determine the extent of these nonlinearities. Steady state data was collected from the engine simulation for 1260 operating points. These 1260 operating points were obtained by varying PLA, DTAMB, ALT, and XM. PLA was varied over a range from 13-77. Temperature was deviated ±20 degrees Fahrenheit from standard conditions in increments of 10 degrees. The variations for altitude and Mach number are shown in Figure 8. These variations in altitude, ALT, and Mach number, XM, are typical for a commercial turbofan engine. Figures 9 and 10 show the variations of WF, MMVFB, and XNH for the 1260 steady state operating points. Note that fuel flow is almost a quadratic function of the main metering valve position, which one might expect for the choked flow of a valve with a flow area that is proportional to the square of the valve position. The corrected rotor speed is plotted as a function of the corrected fuel flow in Figure 10 for the 1260 operating points. Equation (5-8) gives the relationships used to calculate the corrected terms.

$$\text{theta} = T_2 / T_{std} , \quad T_{std} = 518.7°F \qquad (5)$$

$$\text{delta} = P2 / P_{std} , \quad P_{std} = 14.67 \text{ psia} \qquad (6)$$

$$XNH_c = XNH / \sqrt{\text{theta}} \qquad (7)$$

$$WF_c = WF / \text{delta} / \sqrt{\text{theta}} \qquad (8)$$

The curve shown in Figure 10 is very smooth and we assume that this is by design. There is probably a curve like Figure 10 that is part of the engine control system schedule. For this program we did not have access to the details of control system. The fault detection scheme was developed separately and the control system was considered to be a "black box".
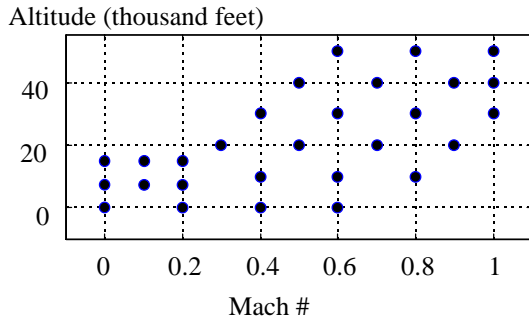
Altitude (thousand feet)

Figure 8 Operating Range of Engine and N.N.'s
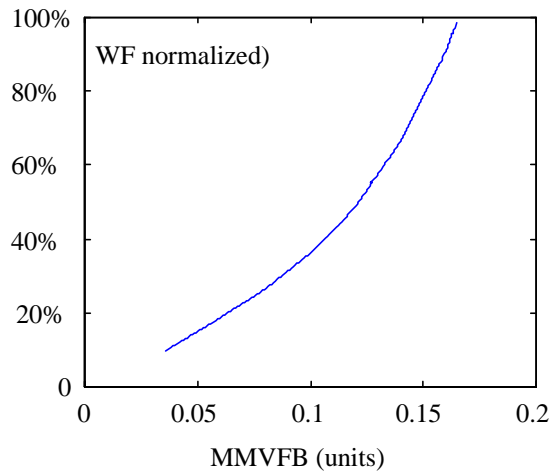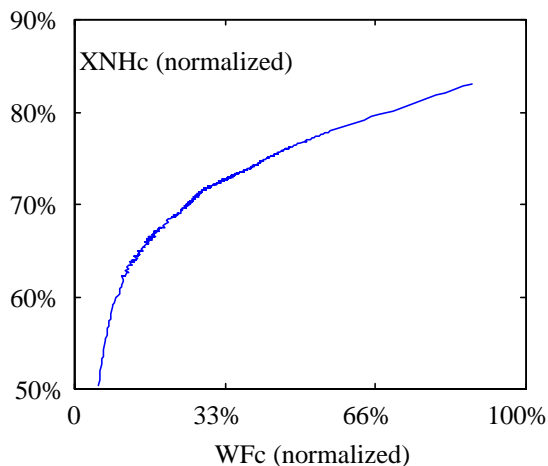

Figure 9 MMVFB vs WF (N.N. #1)

Figure 10 XNH$_c$ vs WF$_c$ (N.N. #2)

The original 1260 operating points were combined with two perturbations around these 1260 operating points to form a total set of 3780 steady state operating points that were used to evaluate the steady state observer performance. Also, the dynamic performance of the

observer was evaluated using a PLA "SLAM" from PLA=21 to 75 and back to 21. This maneuver was performed at 420 different operating points obtained from the same variations in ALT, XM and DTAMB used to generate the steady state estimates. Table 1 lists the worst case steady state and transient performance of the estimator. Note that worst case is conservatively defined by the largest absolute percent error, where percent error is defined in equation (10) as:

$$\%\,\text{error} = \frac{\text{"true"} - \text{"estimate"}}{\text{"true"}} * 100\% \tag{10}$$

Table 1 also lists the worst case estimate error for WF and XNH. The absolute error corresponds to the point where the worst case percent error was calculated. It can be observed that while a 20.2% dynamic error was the largest percent error recorded in MMVFB, this corresponds to an absolute error of only 0.009 out of a full scale of 0.165, which is only 5% of full scale.

Table 1 Worst Case Estimate Errors			
	MMVFB (inches)	WF (lbm/hr)	XNH (rpm)
Range	0.13	2240	5400
Absolute steady state error	0.006	291	290
Percent steady state error (%)	8.6	16.9	2.0
Absolute dynamic error	0.00935	143	1338
percent (%) dynamic error	20.2	28.0	12.1

Note the steady state and dynamic errors are not calculated at the same point. Since the worst case percent error is based on the 'true' error, it is possible for the absolute steady state error to be larger than the absolute dynamic error in Table 1.

The steady state error for XNH is very good. The steady state error for MMVFB of 8.6% is higher than we would have liked, although we believe this error can be improved by modifying the training method. One approach would be to combine the two neural networks into one network, and including P2 and T2 as inputs to the network making this a new, three input, one output network. This would require the network to learn the process of calculating the corrected values. Another approach would be to combine the two networks through an intermediate calculation of corrected variables and then to modify the training algorithm so that both networks could be trained simultaneously. In the current approach these neural networks were trained separately. Both of these approaches would avoid the problem of the accumulation of error when cascading two neural networks.
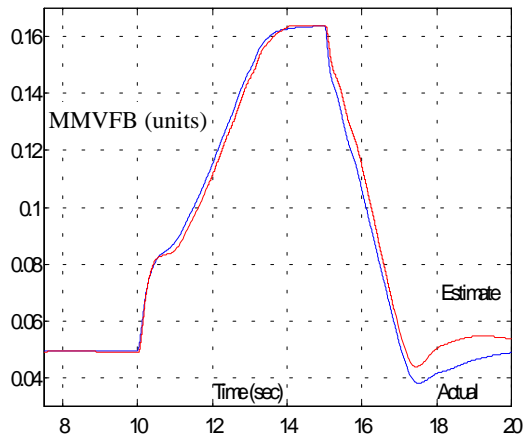
Figure 11 Transient Response of MMVFB Observer

Figure 11 shows the response of MMVFB and estimated MMVFB during the PLA response (PLA=21-75-21) for the case where XM=0, ALT=0, DTAMB=0. In this particular example the estimated MMVFB value was fed back to the control system in place of the actual value to demonstrate the closed-loop stability using the estimated value of MMVFB. This response is typical of the estimator transient response. An important use of such an estimator would be to provide a third vote in a dual redundant FADEC system so that if one of two MMVFB measurements did fail, the MMVFB estimator would be able to cast the tie breaking vote, allowing engine operation to continue normally. This would also allow maintenance to be scheduled to address the faulty sensor at an appropriate time, thus improving the dispatchability of the aircraft.

The two neural networks used here were very simple. The first network is a two layer, 1-4-1 network with a total of 4 nonlinear nodes (the output layer is linear) and a total of 13 variables (8 weights, 5 biases). Network #1 takes less than 10 lines of "C" code to

program. The second neural network that estimates corrected high rotor speed from corrected fuel flow is a 1-3-1 two layer network with a total of ten variables (6 weights, 4 biases). The output layer is linear. These two networks are simple enough that it would be possible to have online training to keep the steady state estimates accurate. Keeping the dynamic response accurate would take more effort, but the problem is small enough that it would be manageable with current computer hardware.

AANN Sensor Validation of XNL, XNH

An Auto-Associative Neural Network is used to recover from sensor failures in the two rotor speed measurements, XNL and XNH, both of which are used in the engine control loop. Several different network architectures were attempted and the bottle-neck layer size was varied from 1 to 4. The goal of changing the network size was to minimize the total number of nodes in the network while still obtaining accurate estimates. The initial goal for this phase of training was to estimate the measured values within 4% of the "true" value. Figure 12 shows the 7-10-4-10-7 auto-associative neural network and the sensor variables used as inputs to this network. This network has 31 neurons, 220 weights and 31 biases to be adjusted during the training process. In Figure 12, the subscript "c" stands for a corrected value.

Network Training

Besides the selection of the analytically redundant sensors as previously described, network training plays a critical role in the success of the sensor validation scheme. There are three steps in the network training.
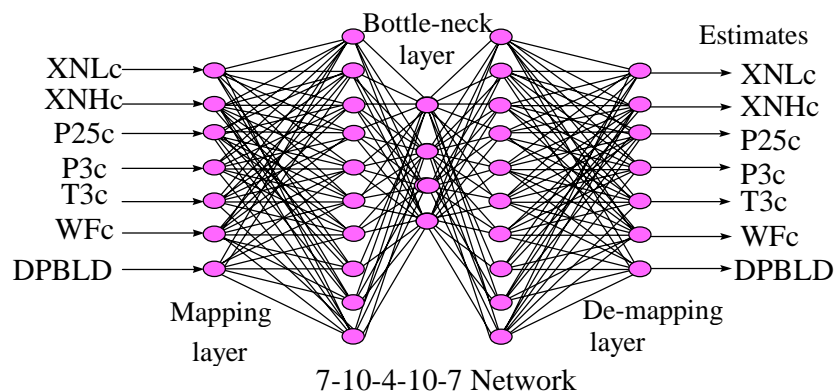

7-10-4-10-7 Network

Figure 12 Auto-Associative Neural Network Schematic for Turbofan Engine Estimates

The first step is the generation of the training data from data collected from the previously described 1260 operating points defined by the variations in ALT and XM (in Figure 8) and PLA and DTAMB. Each operating point contains a corresponding set of sensor values containing the steady state measurement of [XNL, XNH, P25, P3, T3, WF, DPBLD].

The second step is the actual training. The goal is to train the Auto-Associative Neural Network so that:

1) the network output vector matches the input vector at each operating point; and

2) the network output vector shall be insensitive to a single sensor deviation from its "normal" reading.

In this study, a back-propagation algorithm is used to adjust the weights of the network so that the network output will return the desired sensor measurements for both the normal data set and the simulated failed sensor data set. Because the nature of the desired neural network is not a simple functional map, it is necessary to modify the training procedures in order to achieve the best results. The following factors were used for the neural network training:

1) Normalization of the sensor data: All the sensor data are normalized and scaled to have a value between -1 and +1. This is to assure that all sensors will have approximately the same sensitivities.

2) Training with normal data set: The network is first trained with the normal data set to quickly train the network to perform under the normal conditions. By normal we mean, "no fault".

3) Training with simulated failed sensor data set: A sensor failure is simulated by adding a random number to a selected sensor reading in the normal data set. There are two methods of training using the failed sensor data. The first method generated a complete training set with only one failed sensor on each measurement set. It was found that this type of training tended to be slow and sometimes it was difficult to achieve the desired results. The second training method varied which sensor was failed within the training set. In this case, a sensor was randomly selected and random biases were added to the sensor reading. The goal of training with data containing faults is to adjust the weights so that the neural network will minimize the effect of the bad sensor readings by using other sensors to provide a good estimate. In this training, it was also found to be helpful to freeze the first layer weights connected to the node of the bad sensor during the back-propagation weight adjustment. This prevented the failed sensor from being totally

ignored, but required a modification to the standard back propagation scheme.

4) Step size and momentum term: It was found that the momentum term in the training does not improve the training result because of the batch training process. The step size selection depends on the size of the training batch. In this case, it was selected to be in the order of $1.0*10^{-5}$ because of the large batch size of the training set.

AANN FDIA Results

The Auto-Associative Neural Network was combined with simple error threshold logic to construct a fault detection, isolation and accommodation system. The focus here is on the neural network estimates and not on the threshold logic. A comparison was made between the AANN input and output. If the output exceeded a prescribed level, a fault was said to have occurred. One at a time faults are easily detected and isolated because the neural network provides a good estimate of the actual sensor value. While the detection and thresholding logic play an important role in the resulting transient response during the switch from a faulty sensor to an estimated value, in the following we focus primarily on the estimate accuracy of the auto-associative neural network. We will show a simulated slow soft fault, then a fast soft fault, and finally we will consider a hard fault.

In Figure 13, the slow soft fault was simulated for the low rotor speed, XNL, by adding a bias to the sensed value of XNL. The negative bias value was a function of time. Note that because the control system was trying to regulate XNL, the sensed value of XNL remained constant and the actual value increased. The actual value of XNL increased because the controller was increasing the fuel flow to compensate for the negative ramp bias in the sensed value of XNL to hold it constant. Once the measured and estimated values of XNL differed by more than the preset threshold value of 1000 rpm, the measured value for XNL was replaced by the value of XNL estimated by the AANN. A step response can be observed starting at time equal to 15 seconds when the controller began to regulate XNL based on the XNL estimate. The actual, measured, and estimated value of XNL were plotted in Figure 13 along with the XNL fault detection flag which shows when the fault was detected by exceeding the threshold value. Note in Figure 13 at a time of 30 seconds that there remains a bias or offset in the XNL estimate. This is due to the AANN estimation error at this particular operating point. This value is approximately 100 rpm at a nominal value of 7000 rpm.
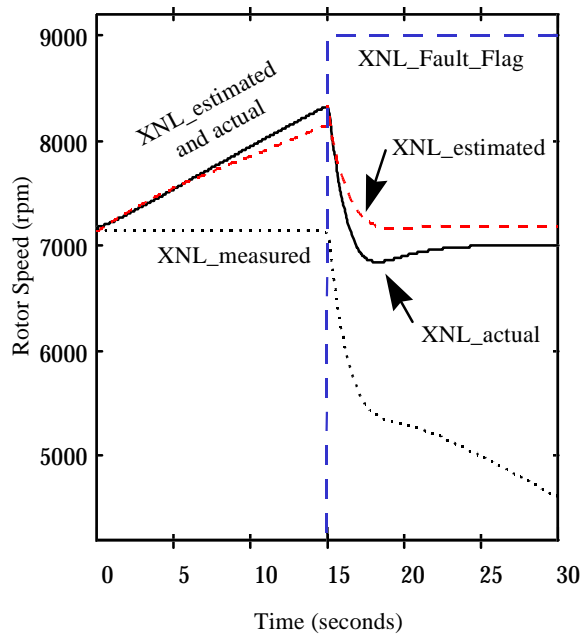
Figure 13 Simulated XNL Slow Soft Fault by Adding a Ramp Bias Value to XNL_measured
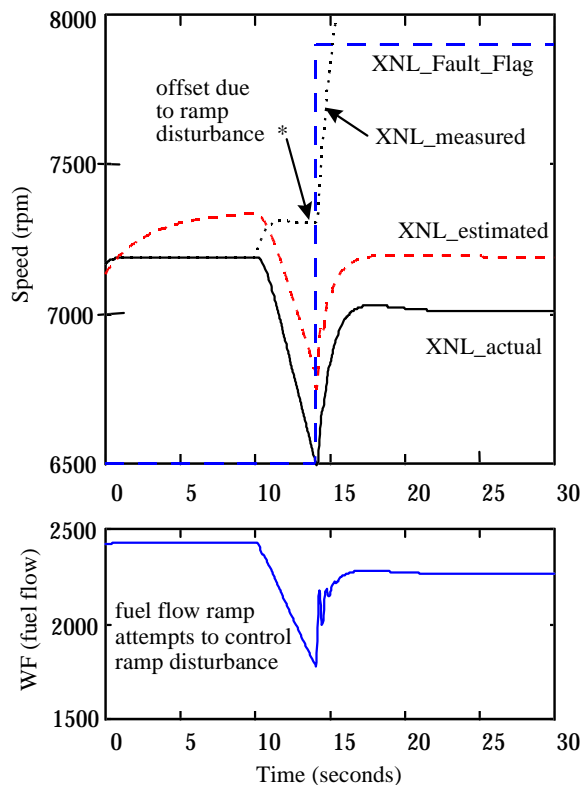

Figure 14 Simulated XNL Fast Soft Fault by Adding a Ramp Bias Value to XNL_measured

The fast soft fault was simulated by adding a fast changing ramp value to the measured value of XNL. The response during this transient is shown in Figure 14 and it is similar to the response shown in Figure 13, except faster and in the opposite direction. The threshold value for this particular example was set to 500 rpm. It can bee seen the difference between the measured and estimated values of XNL exceeds the 500 rpm error threshold at approximately 14 seconds. Also note that the bias started at time equal to ten seconds. Between 10 and 14 seconds the control system is trying to compensate for the ramp increase disturbance in the measured value of XNL by decreasing the fuel flow (see the bottom plot in Figure 14). The controller responses results in a constant offset or bias which is the typical response for a type I control system [9]. As before, once the fault threshold value was exceeded, the control system went through an XNL step response and then continues to regulate on the estimated value of XNL.

The hard fault was simulated for the high rotor speed by adding a large bias term to the measured value of XNH. In Figure 15 the bias value was added at ten seconds. The fault was detected immediately because the fault threshold for XNH was exceeded. The control system switched to the estimated value of XNH. The control system handled the transient as it adjusted to the estimated value of XNH. We do not know the control law, but we believe that the transient in Figure 15 is different from the responses shown in Figure 13 and 14 for XNL because of how XNH is used in the control system. The controller recovered in about two seconds.
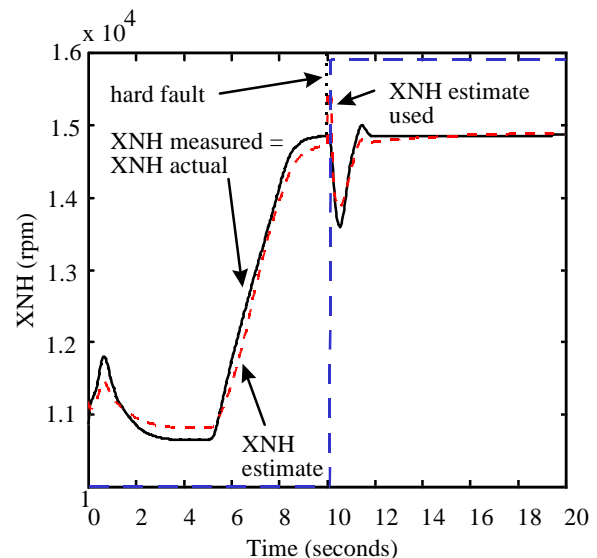

Figure 15 Simulated XNH Hard Fault by Adding a Large Bias Value to XNH_measured

Comments

The results presented here indicate that neural networks can play a role in sensor fault detection, isolation, and accommodation. Using neural networks for nonlinear functional approximation is reasonably straight forward. The results of the auto-associative neural network presented here were good. The AANN was able to detect faults in XNL and XNH using a simple thresholding scheme and the estimated value could be used within the closed loop control system. However, training the network with the fault modified training set was laborious and time consuming. The network size was not huge, but when combined with the fault training set the training times became significant. Training times in excess of 24 hours on a Pentium PC platform were not uncommon. The network had 251 weights and biases to adjust. The fault training set consisted of 1260*(# sensors)*(#simulated faults) training cases. The training times were substantial partially because the 1260 operating points covered a wide operating range. Several different auto-associative neural networks were designed. They were of different structures and used different variables. The training of the neural network was delicate and sensitive to the learning rate and required an experienced hand. In most cases the weights obtained for a given neural network structure were not unique, i.e., different sets of weights produce similar results.

Summary

We have presented two approaches to the sensor validation problem. Two different types of neural networks were used: functional approximation and auto-associative neural networks. The functional approximation neural networks were used as part of a nonlinear, model-based approach to analytical redundancy. The auto-associative neural network was used as part of an FDIA scheme that acted like a fault filter and only required the addition of some thresholding logic. The results that were presented show that neural networks can play a role in fault detection. We believe more work is required to obtain a time-efficient training method for the auto-associative neural network approach.

References

1) Merrill W., Delaat J., Bruton W., "Advanced Detection, Isolation, and Accommodation of Sensor Failures: Real-Time Evaluation," AIAA J. of Guidance, Control and Dynamics, Vol. 11, No. 6, 1988.

2) Menke T.E., Maybeck P.S., "Sensor/Actuator Failure Detection in the Vista F-16 by Multiple Model Adaptive Estimation", IEEE Trans. On Aerospace and Electronic Systems, Vol 31, No. 4, Oct. 1995.

3) Guo T.H., Musgrave J., Lin C., "Neural Network Based Sensor Validation for Reusable Rocket Engines", 1995 American Control Conference, Seattle, WA, June, 1995.

4) Mattern, D., Jaw L., Guo T.H., Graham R., McCoy W., "Simulation of an Engine Sensor Validation Scheme using an Auto-Associative Neural Network", AIAA 97-2902, presented at the 1997 Joint Propulsion Conference, Seattle, WA USA.

5) Kramer M.A., "Autoassociative Neural Networks", Computers & Chemical Engineering, Vol. 16, No. 4, pp 313-328, 1992.

6) Kerr L.J., Nemec T.S., Gallops G.W., "Real-Time Estimation of Gas Turbine Engine Damage Using a Control-Based Kalman Filter Algorithm", Trans. of the ASME, Journal of Engineering for Gas Turbine and Power, April 1992, Vol 114, p 187.

7) Adibhatla S., Lewis T., "Model-Based Intelligent Digital Engine Control", AIAA 97-3192, 33rd Joint Propulsion Conference, July 1997, Seattle, WA, USA.

8) Lin S., Lee C.M., "Multivariable Control of the J-85 Turbojet Engine for Full Flight Envelope Operation", AIAA Journal of Guidance, Control, Dynamics, Vol. 19, #4, 1996, p913.

9) Ogata, K., Modern Control Engineering, Prentice-Hall, Inc., 1970, p. 286.

REPORT DOCUMENTATION PAGE

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 1998	Technical Memorandum

4. TITLE AND SUBTITLE

Using Neural Networks for Sensor Validation

6. AUTHOR(S)

Duane L. Mattern, Link C. Jaw, Ten-Huei Guo, Ronald Graham, and William McCoy

5. FUNDING NUMBERS

WU–519–30–53–00

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio 44135–3191

8. PERFORMING ORGANIZATION REPORT NUMBER

E–11258

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546–0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA TM—1998-208483
AIAA–98–3547

11. SUPPLEMENTARY NOTES

Prepared for the 34th Joint Propulsion Conference cosponsored by AIAA, ASME, SAE, and ASEE, Cleveland, Ohio, July 12–15, 1998. Duane Mattern and Link C. Jaw, Scientific Monitoring, Inc., Tempe, Arizona; Ten-Huei Guo, NASA Lewis Research Center; Ronald Graham and William McCoy, Allison Engine Company, Indianapolis, Indiana. Responsible person, Ten-Huei Guo, organization code 5530, (216) 433–3734.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified - Unlimited
Subject Category: 06   Distribution: Nonstandard

This publication is available from the NASA Center for AeroSpace Information, (301) 621–0390.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This paper presents the results of applying two different types of neural networks in two different approaches to the sensor validation problem. The first approach uses a functional approximation neural network as part of a nonlinear observer in a model-based approach to analytical redundancy. The second approach uses an auto-associative neural network to perform nonlinear principal component analysis on a set of redundant sensors to provide an estimate for a single failed sensor. The approaches are demonstrated using a nonlinear simulation of a turbofan engine. The fault detection and sensor estimation results are presented and the training of the auto-associative neural network to provide sensor estimates is discussed.

14. SUBJECT TERMS

Neural nets; Sensors; Fault tolerance

15. NUMBER OF PAGES

18

16. PRICE CODE

A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	